Claims:

1. A system comprising:

a pipelined central processing unit with associated native program counter ; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator including a reissue buffer, the reissue buffer adapted to store converted native instructions issued to the CPU along with associated native program counter values, the system is such that when the CPU returns from an interrupt, the reissue buffer examines the program counter value to determine whether to reissue a stored native instruction value.

2. The system of Claim 1, wherein the stack-based instructions are Java bytecodes.

3. The system of Claim 1, wherein the hardware accelerator is not flushed upon an interrupt.

4. The system of Claim 1, wherein the hardware accelerator includes a native PC monitor which monitors the value of the native PC.

5. The system of Claim 4, wherein the native PC monitor enables the hardware accelerator when the native program counter is within a hardware accelerator program counter range.

6. The system of Claim 5, wherein an interrupt causes the native PC to leave the hardware accelerator program counter range, causing the hardware accelerator to

stall.

7. The system of Claim 6, wherein the return from interrupt causes the native PC to go back within the hardware accelerator program counter range, enabling the hardware accelerator.

8. The system of Claim 1, wherein the reissue buffer provides stored converted instructions when the system returns from an interrupt.

9. The system of Claim 1, wherein at least portions of the hardware accelerator are part of the CPU.

10. A system comprising:

a central processing unit with associated native program counter ; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator including a native program counter monitor, the native program counter monitor checking whether the native program counter is within a hardware accelerator program counter range, wherein when the native program counter is within the hardware accelerator program counter range, the hardware accelerator is enabled, converted native instructions are sent to the CPU from the hardware accelerator, and the native program counter is not used to determine instructions to load from memory.

11. The system of Claim 10, wherein an interrupt causes the native PC value to leave the hardware accelerator program counter range, causing the hardware accelerator to be stalled.

12. The system of Claim 11, wherein a return from interrupt causes the native PC value to be within the hardware accelerator program counter range, re-enabling the hardware accelerator.

13. The system of Claim 12, further comprising a reissue buffer, the reissue buffer adapted to store converted native instructions issued to the CPU along with associated native program counter values, the system being such that when the CPU returns from interrupt, the reissue buffer examines the program counter value to determine whether to reissue a stored native instruction value.

14. The system of Claim 10, wherein the native program counter monitor further includes a unit to cause a jump in a native program counter to a start portion of the hardware accelerator program counter range.

15. The system of Claim 10, wherein at least portions of the hardware accelerator are part of the CPU

16. A system comprising:

a central processing unit; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator including a microcode stage, the microcode stage including a microcode memory, the microcode memory output including a number of fields, the fields including a first set of fields corresponding native instruction fields and control bits field that affects the interpretation of the first set of fields by microcode controlled logic to produce a native instruction.

17. The system of Claim 16, wherein the microcode stage includes a microcode address logic portion and a microcode memory portion.

18. The system of Claim 17, wherein the microcode address logic includes logic to step through addresses so that multiple native instructions can be produced from fewer stack-based instructions.

19. The system of Claim 16, further including a reissue buffer, the reissue buffer adapted to store converted native instructions issued to the CPU along with associated native program counter values, the system being such that when the CPU returns from interrupt, the reissue buffer examines the program counter value to determine whether to reissue a stored native instruction value.

20. The system of Claim 16, further comprising a native program counter monitor, checking whether the native program counter monitor is within a hardware accelerator program counter range.

21. The system of Claim 16, wherein at least portions of the hardware accelerator are part of the CPU

22. A system comprising:

a central processing unit; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to receive stack-based instructions, the hardware accelerator including a microcode generating unit adapted to receive stack-based instructions and to produce therefrom microcode instructions, the hardware accelerator also including microcode interpretation logic adapted to receive the microcode and to produce therefrom native instructions which are sent to the central

processing unit.

23. The system of Claim 22, wherein the microcode includes fields for native instruction portion and fields for additional control bits.

24. The system of Claim 23, wherein the control bits control the interpretation of fields for the native instruction.

24. The system of Claim 22, further comprising a decoding unit, the decoding unit being a part of the microcode generating unit, the decoding unit producing additional control signals which are provided to the native instruction composer unit to produce the native instructions.

25. The system of Claim 22, further comprising a stack manager unit used to control which elements in the stack are stored within the register file and send data which is used by the native instruction composer unit to compose the native instructions.

26. The system of Claim 22, wherein at least portions of the hardware accelerator are part of the CPU

27. A system comprising:

a central processing unit; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing an indication of the top of operand stack pointer, the top of operand stack being stored and updated in hardware, wherein when more than one stack-based instruction is

translated into a single register-based instruction, the top of stack pointer is modified so as to reflect the effects of each register-based instruction, stack based instruction and instruction level parallelism.

28. The system of Claim 27, wherein at least portions of the hardware accelerator are part of the CPU.

29. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing an indication of the depth count of the portion of the operand stack stored in the central processing units register file, the depth count being updated during the translation process.

30. The system of Claim 29, wherein at least portions of the hardware accelerator are part of the CPU.

31. A system comprising:

a central processing unit; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing an indication of the depth count of the portion of the operand stack stored in the central processing units register file, the depth count being updated during the translation process, the hardware accelerator checking to see if the stack depth is below a minimum or above a maximum depth, wherein if the depth is below the minimum

depth the hardware accelerator generates load instructions to load operand stack data from external memory to the register file, and wherein if the depth is above the maximum depth the hardware accelerator generates store instructions to move operand stack data from register file to the external memory .

32. The system of Claim 31, wherein at least portions of the hardware accelerator are part of the CPU.

33. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing an indication of the operands and variables stored in the register file of the central processing unit, the stored indications being used during the conversion process and being updated by the hardware accelerator.

34. The system of Claim 33, wherein at least portions of the hardware accelerator are part of the CPU.

35. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing an indication of the variable base and top of operand stack in the memory, the stored indications being used by the hardware accelerator to compose loads and stores of variables and operands in and out of the register file.

36. The system of Claim 35, wherein at least portions of the hardware accelerator are part of the CPU.

37. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing at least the top four (4) entries of the operand stack in the native CPU register file as a ring buffer, the ring buffer maintained in the accelerator and operably connected to a overflow/underflow unit.

38. The system of Claim 37, wherein at least portions of the hardware accelerator are part of the CPU.

39. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, the hardware accelerator storing Java variables in the native CPU register file and an indication of which variables are in the native CPU register file.

40. The system of Claim 39, wherein at least portions of the hardware accelerator are part of the CPU.

41. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator composes native instructions based on the availability of variables and operands in the native CPU register file.

42. The system of Claim 41, wherein at least portions of the hardware accelerator are part of the CPU.

43. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator marks the variables in the native CPU register file as modified when updated by the execution of Java byte codes.

44. The of claim 43, wherein the hardware accelerator copies the variables marked as modified to the system memory for some bytecodes.

45. The system of Claim 44, wherein at least portions of the hardware accelerator are part of the CPU.

46. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator

issues native load instructions when a variable is not present in the native CPU register file, the memory address being computed by an ALU in the hardware accelerator.

47. The system of Claim 46, wherein at least portions of the hardware accelerator are part of the CPU.

48. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator composes native instructions wherein the native instructions operands contains at least two native CPU register file references where the register file contents are the data for the operand stack and variables.

49. The system of Claim 48, wherein at least portions of the hardware accelerator are part of the CPU.

50. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator generates a new Java PC due to a "GOTO" or "GOTO_W" byte code.

51. The system of Claim 50, wherein at least portions of the hardware accelerator are part of the CPU.

52. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator generates a new Java PC due to a "JSR" or "JSR_W" byte code, computes the return Java PC and pushes the return Java PC on to the operand stack.

53. The system of Claim 52, wherein at least portions of the hardware accelerator are part of the CPU.

54. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator sign extends the SiPush and Bipush byte codes and appends to the immediate filed of the native instruction being composed.

55. The system of Claim 54, wherein at least portions of the hardware accelerator are part of the CPU.

56. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator sign extends the SiPush and Bipush byte codes and made available to be read by the native

CPU.

57. The system of Claim 56, wherein at least portions of the hardware accelerator are part of the CPU.

58. A system comprising:

a central processing unit with associated register file; and

a hardware accelerator operably connected to the central processing unit, the hardware accelerator adapted to convert stack-based instructions into register-based instructions native to the central processing unit, where the hardware accelerator increments the Java PC within the hardware accelerator by generating an increment value based on the number of byte codes being disposed.

59. The system of Claim 58, wherein at least portions of the hardware accelerator are part of the CPU.